

Structuring State

Self-Review Questions

Self-review 4.1 What are the values of the variables `a`, `b`, `c` and `d` after the following statements have been executed?

```
a = 1
b = 2
c = a + b
d = a + c
```

`a` will be 1, `b` will be 2, `c` 3 and `d` 4.

Self-review 4.2 How many elements are in the dictionary `someData` after the following code has been executed?

```
someData = {}
someData['cheese'] = 'dairy'
someData['Cheese'] = 'dairy'
```

Self-review 4.3 Given the following assignment:

```
items = ((3, 2), (5, 7), (1, 9), 0, (1))
```

without writing a Python program or using the Python interpreter, answer the following:

1. What value is returned by `len (items)`?
2. What is the value of `items[1]`?
3. **print** `items[2][0]` prints 1 but **print** `items[3][0]` causes a `TypeError`. Why is this given that `items` has more than 3 elements?
4. For `items[x][y]`, what values of `x` and `y` cause the expression to evaluating to 9?
5. It might at first appear that the type of `items[0]` and the type of `items[4]` are the same.
 - (a) What are the types of these two expressions?
 - (b) Why is the type of `items[4]` not `tuple`?
 - (c) If it was intended that `items[4]` be a `tuple`, how should the assignment be altered, without adding any new values?

Self-review 4.4 What is the output of the following code?

```
a = 1
def f():
    a = 10
print a
```

The code will print 1 to the console.

Self-review 4.5 Explain why the code in the previous question did *not* display 10.

Because the scope of the line `a = 10` is the method `f`, which is never called.

Self-review 4.6 Referring to the coffee/cola vending machine state transition diagram (page ??), what happens if the consumer puts three tokens into the machine without pressing any buttons?

Self-review 4.7 Which of the following are types mutable and which are immutable: int, list, float, tuple, str.

All the types listed are mutable except for tuple and str.

Self-review 4.8 What happens when the following statement is executed as the first statement of a program?

```
x, y, z = 1, 2, x ** 3
```

There is an error – the exception `NameError` is raised. All the expressions on the right-hand side are evaluated before any assignments are done so `x` is being used before it is assigned.

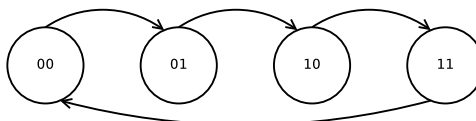
Self-review 4.9 For each of the following lines, using substitution, hand-evaluate the following sequence:

```
x, y, z = 1, 2, 3
z, y, x = x, z, y
x, y, z = (z + 1), (x - 2), (y * 3)
y, z, x = (y / x), (x * y), (z ** x)
```

Self-review 4.10 Draw the state space of the following devices:

1. An MP3 or Ogg Vorbis player.
2. A microwave oven.

Self-review 4.11 Draw a state transition diagram for a two-bit binary counter. A binary counter is a digital circuit that has a clock input and an output that gives the number of clock cycles that have been counted. A two-bit binary counter only has two output connections, which can have the value of either 1 or 0. This means that a two-bit counter can count from 0 to $2^2 - 1$ and back to 0 again.



Self-review 4.12 Draw a state transition diagram for a PIN (Personal Identification Number) recognizer. If the user enters '1' followed by '8' followed by '5' followed by '0' followed by 'enter', the machine should enter a state called 'accept'. If the user enters any other combination of numbers, followed by 'enter' the machine should enter a state called 'reject'. Is this program a lexer?

Self-review 4.13 The implementation of cipherCharacter on page ?? has a bug in it even though this bug may never appear in normal use. What is the bug and how can it be fixed?

The problem is that there is no wrap around, and the chr function requires its parameter to be in range(256).

Self-review 4.14 What are the types of variables a and b in the following code:

```
a = (1)
b = (1,)
```

Self-review 4.15 What is the types and value of y after executing the following code:

```
x = [1, 2, 3, 4, 5]
y = x[:]
```

Both x and y are lists. They have the same value, which is [1, 2, 3, 4, 5].

Programming Exercises

Exercise 4.1 Examine the following code:

```
a = 10
b = 20
print "Before swapping a=%d, b=%d" % (a, b)
#Swap values
a = b
b = a
print "After swapping a=%d, b=%d" % (a, b)
```

1. Try running this small program. What are the values of a and b after the swap?
2. Why is the result not a having the value 20 and b having the value 10 as appears to be the intention from the use of the word swap in the code?
3. Modify the code so that the final result is a having the value 20 and b having the value 1 using simultaneous assignment to swap the values.
4. Many languages, such as C, C++, Java, do not have simultaneous assignment. Modify the program so that it successfully swaps the two values without using simultaneous assignment.

1. Both `a` and `b` have the value 20.
2. Because the line `a = b` sets `a` to 20, leaving `b` as it was.
3. `a, b = b, a`
 4. `a = 10`
`b = 20`
`print "Before swapping a=%d, b=%d" % (a, b)`
`# Swap values`
`temp = a`
`a = b`
`b = temp`
`print "After swapping a=%d, b=%d" % (a, b)`

Exercise 4.2 This question is based on a function for stepping through a representation of a pack of cards. The exact nature of the cards isn't important. We use a list to represent pack and in our example we use numbers between 1 and 5 as cards. They could equally be numbers and suits as tuples, strings of character names, etc. Here's the function:

```
# A hand of cards
cards = [ 1 , 5 , 3 , 4 , 2 , 3 , 2 ]

def nextCard ( cards ) :
    next = cards[0]
    newHand = cards[1:] + [ cards[0] ]
    return next , newHand
```

- What is the type of values returned by this function?
- Describe in words what the function does.
- It would be a simple matter for this function to alter the input list so that the first element becomes the last. This would simplify the return value, which is currently two items. Why might this be considered a poorer solution than the function as it is now?
- Write a loop that calls this function repeatedly, printing out the card from the top of the deck each time.
- Using the given function and your answer to the previous question as a starting point, write a program that checks a deck of cards for consecutive identical pairs (a 2 followed by a 2, for example). If a pair is found, a message should be displayed.
- What happens if the input list is empty? Correct the function so that it returns (`None` , `[]`) when an empty list is used as input.

Exercise 4.3 Write a program to create a frequency table using a dictionary from a set of input values. The program should repeatedly ask the user for an input. The input could be anything at all. As each input is gathered, see if it exists as a key in the dictionary. If it does not exist, associate the input as a key with the number 1. If it does exist, add one to the value already associated with it and add it back to the dictionary. If the value associated with this key is now greater than or equal to 3, print a message that looks something like "Dear user, you have entered the word 'discombobulated' 3 times!"

Frequency tables such as this are often used for storing histogram data. We will look at actually drawing histograms later (Chapter ??). Frequency tables are also used in some sorting algorithms.

Hint: You will need an infinite loop and to use the `raw_input` function.

Hint: `someDict.has_key ('a key')` returns a Boolean value indicating whether the dictionary `someDict` has already got a value associated with the key 'a key'.

```
def frequency () :
    d = {}
    while True :
        user = raw_input ( 'Please enter the next value or QUIT to finish: ' )
        if user == 'QUIT' : return
        elif d.has_key ( user ) :
            d[user] += 1
            if d[user] > 2 : print 'Dear user, you have entered the value', user, d[user], 'times!'
        else :
            d[user] = 1
```

Exercise 4.4 Write the Python program that manages the state of the two-bit binary counter from Self-review 4.11

Exercise 4.5 The lexer in Section ?? can only recognize non-negative integers. Improve the program by adding code to deal with negative numbers.

```
def getIntegerValue ( buffer ) :
    isNegative = False
    if buffer[0] == '-' :
        isNegative = True
        buffer = buffer[1:]
    for i in range ( len ( buffer ) ) :
        if not buffer[i].isdigit () : break
    if isNegative:
        return -1 * int ( buffer [(i+1)] )
    else:
        return int ( buffer [(i+1)] )
```

Exercise 4.6 Create a program for storing a week's worth of rainfall data. Use a list to store each day's value, entered sequentially by the user. When an entire week has been input, display the days with the minimum and maximum rainfall.

```
import math

def rainfall () :
    days = [ 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun' ]
    rain = []
    for i in range ( 7 ) :
        r = input ( 'Please enter the rainfall for ' + days[i] + ': ' )
        rain.append ( r )
    minimum = rain[0]
```

```

maximum = rain[0]
for i in rain :
    if i < minimum : minimum = i
    if i > maximum : maximum = i
print 'Min rainfall for this week:', minimum
print 'Max rainfall for this week:', maximum
for i in range ( len ( rain ) ) :
    print 'Rainfall for', days[i], '=', rain[i]

if __name__ == '__main__':
    rainfall ()

```

Exercise 4.7 Extend your answer to the previous exercise so that it also displays the mean and standard deviation of the values. The mean is the sum of all the values divided by the number of values. The standard deviation is the square root of the sum of the squares of the difference between each value and the mean, divided by the number of items.

```

import math

def rainfall () :
    days = [ 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun' ]
    rain = [ ]
    for i in range ( 7 ) :
        r = input ( 'Please enter the rainfall for ' + days[i] + ': ' )
        rain.append ( r )
    # Minimum and Maximum
    minimum = rain[0]
    maximum = rain[0]
    for i in rain :
        if i < minimum : minimum = i
        if i > maximum : maximum = i
    mean = sum ( rain ) / len ( rain )
    sd = math.sqrt ( sum ( [ ( x - mean ) ** 2 for x in rain ] ) / ( len ( rain ) - 1 ) )
    # Print everything to the console
    print 'Min rainfall for this week:', minimum
    print 'Max rainfall for this week:', maximum
    print 'Mean rainfall for this week:', mean
    print 'Standard deviation rainfall for this week:', sd
    for i in range ( len ( rain ) ) :
        print 'Rainfall for', days[i], '=', rain[i]

if __name__ == '__main__':
    rainfall ()

```

Exercise 4.8 Calculating statistics such as maximum, minimum, mean and standard deviation is a common task. Now that you have a program to do it, package the code up into a function that can be used again in future.

Exercise 4.9 A “wobbly” number is one in which the digits alternate between being higher and lower than the preceding one. Here are some wobbly numbers: 19284756242, 90909, 0909. Using what you have learned

about writing lexers, write a function that accepts a list of digits to be checked for wobbliness. If the sequence of digits is wobbly, the function should return **True**, otherwise **False**.

```
def isWobbly ( num ) :
    if num == " " : return False
    elif not ( len ( num ) > 2 ) : return False
    if not ( num[0].isdigit () or num[1].isdigit () ) : return False
    isLower = int ( num[0] ) < int ( num[1] )
    curr = int ( num[1] )
    for i in num[2:] :
        if not i.isdigit () : return False
        d = int ( i )
        if ( curr < d ) == isLower : return False
        elif curr == d : return False
        isLower = curr < d
        curr = d
    return True
```

Exercise 4.10 The following code was written before the programmer had drunk their morning coffee, i.e. they were not properly awake at the time. It is supposed to add 3% to all the values in the variable `salaryScale`. However, there is one very important bug in this code. Find and fix the bug to make sure everyone gets their 3% pay rise!

```
salaryScale = ( 20000 , 21500 , 23000 , 24500 , 26000 , 27500 , 29000 )
for i in range ( len ( salaryScale ) ) :
    salaryScale[i] += salaryScale[i] * 0.03
print salaryScale
```

Tuples are immutable so the code generates a `TypeError` when executed.

Exercise 4.11 Starting with the list:

```
[ 43 , 1 , 2 , 99 , 54 , 78 , 22 , 6 ]
```

write a function to sort the list – i.e. your function should return the list:

```
[ 1 , 2 , 6 , 22 , 43 , 54 , 78 , 99 ]
```

You can use any method you like to sort the list. Good answers to this problem will work on any list, not just the one we've given here.

Challenges

Challenge 4.1 Imagine a fictional land where monetary units aren't based on decimal orders. In this land, we have 3 basic units of currency:

- The Blink. The smallest unit of currency.
- The Hoojim. Worth 12 Blinks.
- The Bung. Worth 20 Hooja (plural of Hoojim) or 240 Blinks.

1. Write a function called `deBung` that accepts an integer representing a number of Bungs and displays the number of Hooja and Blink it is worth. For example, calling the function like this:

```
deBung ( 4 )
```

Will produce the following output:

```
4 Bungs is worth 80 Hoojim or 960 Blinks.
```

2. Write a function called `enBlinkHoojaBung` that takes a number of Blinks and outputs its equivalent in Blink, Hooja and Bung, using the smallest number of coins. Coins in our imaginary land are made of a very heavy metal, so this is important. If the function is called with the value 506, the output should be:

```
506 Blinks is worth 2 Bung, 1 Hoojim and 6 Blinks.
```

You will need the remainder operator (%).

3. Rewrite `enBlinkHoojaBung` so that it returns a tuple of Blinks, Hooja and Bung values instead of writing to the screen. The last example would return (2 , 1 , 6).

Challenge 4.2 Referring back to Exercise 4.2, write a program that uses two hands of cards (two lists) to play a simple game of “snap”. No cards change hands, but when two identical cards are played, a message should be printed.

Challenge 4.3 Write a lexer to recognize floating point numbers.

Hint: If your input is the following string: ‘12345.9876’ what do you have to do to each character in the string to generate your float? How are the numbers before the decimal point different to the numbers after the decimal point? Make sure you have an algorithm that works correctly on paper before you start coding and draw a state transition diagram for the scanning part of your lexer.